Volume 7 (2007)

Interface: The Journal of Education, Community and Values

2-1-2007

Admiral Grace Murray Hopper

Sarah Monda

Admiral Grace Murray Hopper

Admiral Grace Murray Hopper

Posted on **March 1, 2007** by **Editor**

By Sarah Monda

In 1906, a young couple gave birth to a girl, their first child. The child's name was Grace Brewster Murray, born 9 December 1906 in New York, New York. The surroundings were completely normal, it was cold outdoors, the city of New York was preparing for the Christmas celebrations two weeks into the future. In short, nothing hinted that this child would be different from any other girl-child of the time. The rules governing a proper upbringing for young middle class girls in 1906 were strict. Girls were conditioned to develop interests in the domestic arts such as sewing and cooking, playing the piano, or raising a family. Interest in science, mathematics, or sports was considered abnormal for a girl of this time period. However, Grace Murray's childhood was not a typical one, nor was her family a typical family.

Driven by a fear of being widowed, Grace's mother taught herself to be financially literate, imparting her love of mathematics and spatial geometry to young Grace. Her father meanwhile, realizing that as a double amputee due to arterial sclerosis he would not leave much money to his children after his death, encouraged Grace and her younger brother and sister to be as independent as possible. Grace's parents encouraged her insatiable curiosity for mechanical objects and encouraged her to participate in sports and other 'un lady-like' pursuits. [1] This atmosphere would serve to prepare Grace for her destiny as one of the most influential women in U.S. history. Grace Murray would change the world of information technology forever.

As a child, Grace was always fascinated by mechanical items and mathematics. One day she decided to figure out how her alarm clock worked by disassembling and reassembling it. However, once she got it apart she could not get it back together so she proceeded to take apart six more clocks in order to figure out how to put the first back together. When her mother found out what she was doing, Grace was limited to just one clock. [2] This interest in gadgets and mechanical devices would later combine with an expertise in mathematics attained at Vassar College and lead to a lifetime of love for computers and computer programming.

Grace Murray entered Vassar College in 1924 at the age of seventeen with her interest in mathematics in tow. She graduated from Vassar with a BA in Mathematics in 1928 and went on

to study at Yale where she earned a Masters in Mathematics and Physics in 1930. That same year she also married Vincent Foster Hopper, an English instructor at New York University School of Commerce. They would separate in 1940 and become divorced in 1945, never having any children. In 1934 Grace Murray Hopper achieved that rarest of educational heights, a PhD in mathematics from Yale University. [3] After achieving her Masters in 1930, Grace Hopper began teaching mathematics at Vassar College. [4] She would continue in this pursuit until she joined the U.S. Naval Reserve in 1943 and fell in love with the biggest gadget she had ever seen.

In 1943 as a new lieutenant (jg) [5] Grace Murray Hopper arrived at Harvard University's Cruft Laboratories where she immediately encountered a noisy, cumbersome brute officially named the IBM Automatic Sequence Controlled Calculator (IBM-ASCC) but more commonly known as the Harvard Mark I. She was assigned to work with Commander Howard Aiken (USNR) on developing programs for calculating naval ordnance trajectories. At eight feet high, eight feet deep, fifty one feet long and nearly five tons in weight, the Mark I was the first automatic digital calculator in the United States and was capable of performing the four basic arithmetic operations in numbers up to 23 decimals in length. [6] When asked later how she became interested in computing, Grace replied that she "didn't have a choice in the matter. 'I was ordered to the first computer in the United States by the United States Navy, and I reported to the Mark I computer.'" [7] Grace's assignment at Cruft was to work with a group of officers and enlisted men, under Commander Howard H. Aiken, on writing and running programs for the Mark I that could calculate new ballistics charts for the US Navy, new differential equations for the scientists working on the Manhattan project, and new charts for the use of underwater mines. [8] However, despite the fact that the Mark I was run 24 hours a day, seven days a week the process of programming the beast was not very efficient.

Programming the Mark I was an arduous process. Programmers had to learn every relay switch, all 3,300 of them, and what they did. The reason for this was because the programmers needed to know if an error in the computation process was caused by their program or by a malfunction in one of the relays. There was no such thing as software in the sense of permanent programs on the computer. The Mark I had only seventy-two words of memory so lengthy programs could not be stored permanently. As a result, each programmer kept copious notes and records of various programs to facilitate reproducing programs for different jobs. Each program also had to be broken down into individual functions of addition, multiplication, subtraction, and division and because each programmer wrote his or her own programs, there was no standardization in language. As a result, there were often multiple programs for the same functions, each program being slightly different, depending on who wrote it. [9]

Also, because the Mark I was being run 24 hours a day all year long, and there were not enough people in the lab to work in shifts, mistakes were common. Grace found out that there were two major problems that occurred late in the day. The first was that people could not copy programs correctly; the second was that they could not add. What she found was that symbols like + became 4 or the number 13 would become a B or vice versa. When these slips combined with elementary errors in arithmetic, failures were guaranteed. [10] Hopper realized that people were

not as perfect as the computer for copying and editing, however the Mark I was purely a calculator and lacked the ability to write its own programs. This insight would lead Hopper to seek to remedy the situation for the rest of her career in computer programming.

One of the major problems, as mentioned above, was that there was no standardization in programming language. After the end of World War II, Lieutenant Hopper focused most of her efforts on attempting to solve the language problem. She believed that solving the language problem would solve the problem of human error in programming. Her theory was that if humans were so prone to mistakes, why not make the computer do the work? [11] After joining Eckert-Mauchly Computer Corporation as senior mathematician in 1949, Grace built on her experiences working with the Mark I under Howard Aiken and focused mainly on methods to facilitate the writing and execution of programs and eliminate human error. It would be her work with the BINAC and UNIVAC for which she would become most famous.

The BINAC (Binary Automatic Computer) was the first computer Hopper worked with that had a stored program. It was also programmed in octal—a mathematical system based on eight digits from 0 through 7—what most people believed to be the most efficient way to program a computer. However, this meant that the computer could not understand the decimal system. All conversions from octal to decimal had to be done by hand. Although the BINAC was faster and had a greater capacity for stored information than the Mark I or Mark II, the necessity of relying on humans to convert decimal equations into octal or octal into decimal introduced all kinds of problems for Hopper. Once, after having worked in octal all day, Hopper could not get her checkbook to balance. When she had her brother check it, he realized that occasionally she had used octal instead of decimal. Hopper's solution to this was not that she should learn better octal, but that the computer should learn decimal; the machine should do the work of converting between the two systems. [12] However, before she had the chance to solve this problem with the BINAC, EMCC was bought out by Remington Rand and the UNIVAC I was put into production.

UNIVAC I (Universal Automatic Computer) was designed with commercial applications in mind. Unlike BINAC which, like the Mark I and Mark II, was designed for scientific and mathematical applications, UNIVAC I had to be able to understand alpha-decimal systems because the intention was to market UNIVAC I to businesses for use in settings as varied as insurance firms and mailrooms. Although this solved Grace's problem of working with the octal language, there was still the problem of needing to rely on people to do all of the programming for each operation. Just like the Mark I, UNIVAC I could not understand a standard language. Also, each program was individually generated by the programmers. This meant that there was a lot of unnecessary duplication of effort since most programs used the same basic sets of sub-routines. It also meant that there were a lot of errors generated by the fact that people could not copy correctly.
Hopper believed that she could develop a method by which computers could take over the selection of sub-routines to generate programs themselves based on a standard language. This ability is taken for granted today, but in 1951, the idea of such a compiler was new. [13]

The first step Hopper made toward the creation of the first compiler was to design tapes (the UNIVAC used magnetic tape rather than punch cards) which had the sets sub-routines labeled with three-letter call signs. She then linked individual tapes together, creating the first pseudo-compiler. By entering the proper call sign, the programmer would instruct the computer to retrieve instructions from a series of instruction tapes. This eliminated the need for hand-copying each sub-routine for each program and also made the process of programming a computer much faster. [14]

This pseudo-compiler was the A-0. It was not a true compiler nor was it was it a true "computer language." Rather it was series of specifications that allowed the computer to produce a program. The A-0 allowed programmers to quickly write a mathematical program for one-time execution and get an answer, it did not allow for the computer to generate a complete program on its own from a catalogue of subroutines. The A-0 still relied on "information supplied by the mathematician, under the control of a 'compiling' routine created by a professional programmer, using sub-routines and its own instruction code" to produce a program that would generate the desired results. [15] Because this process still relied on humans and was susceptible to human error, Hopper was still not satisfied. She set out to create a true compiler that could look up the sub-routine itself and generate the program directly. This would allow the program itself to be saved to the 'library' rather than only the sub-routines. "Stated bluntly, the compiling routine is the programmer and performs all those services necessary to the production of a finished program." [16] However there was a problem. When Hopper told others about her idea, they replied that compilers could only do arithmetic, not write programs. Hopper however, set out prove that she could make a computer do anything that could be completely defined and coded. [17]

As the computer industry expanded through the mid 1950s into the private sector (areas outside university or military control), Grace and her colleagues realized that the road to increased sales and support of computers and for computing lay in making them easier to program. Although the A-0 and other early "compilers" were a first attempt at making programming easier, there was still much to be desired. Each machine was programmed separately, with its own codes. This created issues when programmers trained on one computer had to transfer to the use of another. Because she was still a member of the U.S. Naval Reserve, Hopper was keenly aware of the problems that military programmers encountered. Due to their enlistment rotations, military programmers would usually spend two years at one installation, learning the specific codes for that computer. When, after two years, they were transferred to another installation, military programmers had to learn new codes for the new computer. This meant that the programmers were not effective until they learned all the codes. It also created problems due to the perennial enemy—human error—that Grace Hopper sought to eliminate when programmers confused the two different codes. Faced with the necessity to make coding and documenting techniques standard, Hopper began to develop more and more sophisticated versions of the A-0 like the A-1. [18] What resulted was the A-2, the first compiler to have any semblance of a pseudo-language other than machine language. That is to say it was the first to understand non-numerical commands.

Like the A-0 (and the short-lived A-1), the A-2 was designed for use in mathematical operations and problem solving. However, although the A-2 compiler could understand non-numeric commands, it still could not understand English. Following the A-2, Hopper and her team introduced the AT-3 compiler. The AT-3, introduced in 1956 and also known as MATH-MATIC, was an early mathematical language capable of accepting English verbs and mathematical symbols. This compiler was designed to compete with IBM's FONTRAN compiler. However, all of these compilers were still reliant on a programmer's intimate knowledge of mathematical symbols; something the majority of potential customers did not have. As Hopper said, "They wouldn't know a cosine if they met one walking down the street in broad daylight." [19] Because of this reliance on mathematical symbols, there was no way for the MATH-MATIC to be used in applications pertaining to general business needs like cataloging client addresses. To Hopper, the MATH-MATIC, although successful at what it did in the world of math, was not a complete success. In fact her best and most famous compiler would be a B-series compiler that could understand plain English.

The B-0 compiler, also called the FLOW-MATIC compiler, was the first *true* compiler. It was designed to translate English-language instructions and generate its own programs from the library of subroutines. These programs would then be recorded on magnetic tape to be reused as necessary. The FLOW-MATIC made it possible for the UNIVAC I and UNIVAC II computers to "understand" up to twenty statements in English. However, there were still problems. Not everyone could spell very well and spelling errors could cause the computer to stop dead. To solve this, Hopper and her team wrote safe-guards into the FLOW-MATIC that, should the computer encounter an un-recognized word, it would ask for clarification. If the problem happened again, the computer would suggest an alternate spelling and instruct the operator to press the [start] bar if this was a correct suggestion. [20]

FLOW-MATIC was a huge breakthrough. It made computer programming extremely successful and a much easier task to perform. FLOW-MATIC could cope with all of the coding problems inherent in dealing with different computers. If each computer was equipped with FLOW-MATIC, programmers would have no need to learn the specific programming codes unique to that computer, but rather they only needed to know how to use FLOW-MATIC. FLOW-MATIC was also able to understand commands in French or German as well as in English. [21] Despite this success, after the introduction of FLOW-MATIC, Hopper did not cease her quest for simpler, easier to use, compilers and programming languages.

Although the FLOW-MATIC was immensely successful, it was not compatible with other compilers being developed by IBM and other companies. This mish-mash of compilers was not very conducive to establishing Hopper's dream of a universal computer language. To solve the problem, in 1959 Hopper and other representatives of the technology companies asked the Department of Defense to institutionalize meetings between various members of the computer technology industry. These meetings were called CODASYL (Committee on Data Systems Languages). The Committee was sub-divided into three groups which all worked to figure out how to standardize data processing languages. Hopper and Robert W. Bremer of IBM were

selected as special advisors to the executive committee of CODASYL. The members of the executive committee were all people Hopper knew well, and most of them used UNIVAC computers. [22] This was incredibly lucky for Hopper when it came to writing the new common business language.

One of the sub-committees of CODASYL, tasked with investigating the situation and creating a new common language, considered as its models the UNIVAC FLOW-MATIC, IBM's COMTRAN, and the US Air Force's AIMACO. What emerged was an amalgamation of all of these languages—COBOL—in 1960. The bulk of COBOL (Common Business Oriented Language) however, including the format, was based on Hopper's 1958 version of FLOW-MATIC and for the next thirty-five years, versions of COBOL were the standard programming language for large mainframe and small micro- computers alike. [23] What made COBOL so successful was its ability to be run on any machine regardless of manufacturer. On the first day COBOL ran, it was tested on the UNIVAC II: the next day it was tested on the RCA 501. [24] Although COBOL was joined by new versions of FONTRAN in 1966 and JOVIAL (Jules Own Version of the International Algorithmic Language), the languages were standardized by the United States of America Standards Institute so that they could be "'commonly acceptable, competitive, open to extension, and above all useful.'" [25]

Throughout this period, Grace Hopper remained active in the Naval Reserve. In fact, it was often the Navy that gave the best responses to her work on UNIVAC and FLOW-MATIC. In 1952 she had been promoted to Lieutenant Commander. [26] However, as her fame among programming circles spread throughout the 1960s the Navy placed increased demands on her time. She was one of the most experienced programmers in the Navy and was often in demand as a teacher and speaker to new recruits and programmers at various Navy installations. In 1966, Hopper received a letter from the Chief of Naval Personnel informing her that she had served 23 years and was about to turn sixty, the required retirement age for most naval personnel. She was forced to retire, effective 31 December 1966. [27] However she would not stay retired, nor would she stop her work in computers.

The naval career of Admiral Grace Murray Hopper would not end until 1986 when she was just a few months shy of 80 years old. Recalled to active duty in 1967 to facilitate the standardization of all computer programming languages used by the Navy that were not part of the weapons systems, Hopper would not leave active duty until 1986. In 1983, she was promoted to Commodore and to Rear Admiral (l.h.) when the ranks of Commodore and Admiral were merged. [28] At the time of her retirement, Admiral Hopper was the oldest person on active duty in the United States Navy at the age of 80 (Admiral Hyman Rickover had retired in 1982). [29] However, her work with compilers like FLOW-MATIC and COBOL are her greatest claim to fame for without them, the world of computer programming would not be what it is today. Rear Admiral Grace Murray Hopper passed away on 1 January 1992. She is buried at Arlington National Cemetery, Virginia. [30]

Binary, Decimal and Octal Systems

The main feature of each system is its *base* or the number of digits that are used to signify quantities in a number system. In the decimal system, the base is 10 because all numbers are represented by the ten digits 0 through 9. The Binary system has a base of two because it uses only two digits, 0 and 1. 0 through 7 are the digits used in the octal system. [31]

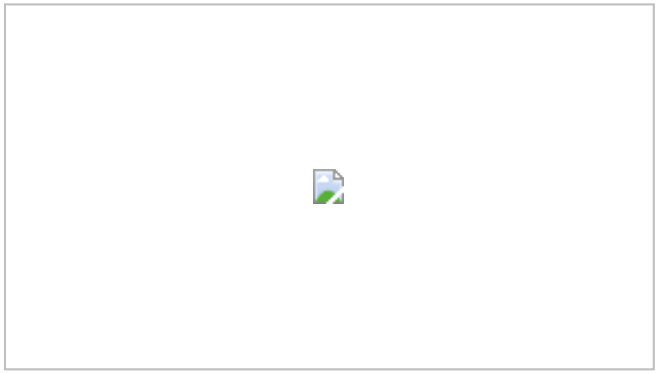
DECIMAL NUMBER	BINARY NUMBER	OCTAL NUMBER
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	10
9	1001	11
10	1010	12
11	1011	13
12	1100	14
13	1101	15
14	1110	16
15	1111	17
16	10000	20
17	10001	21
18	10010	22
19	10011	23
20	10100	24

Coding Sheet for Mark I

OUT	IN	MISC.
74	74	64
2132	74	71
7432	321321	77
321	321	732
	761	7
	321	
21	752	7
7421	7432	7
31	7421	32
3	321	7
321	761	7
31	32	732
3	3	7
872	32	
743		
742	2121	7
1	21	77
2	761	7
2	2	77
132	32	
741	1	
1	761	
	START	87

This is an example of a coding sheet for the Mark I. These numbers would be converted to punch holes in paper tape, which would be inserted into the computer. [32] To create a loop, the ends of the tape would be joined together, creating a physical loop.

Punch cards used by textile mills in looms to determine patterns in cloth were the primary influence for the first program cards used by early computers. Later programming cards would use square holes. The Harvard Mark I used punched paper tape rather than individual cards.



Notes

[1] Elizabeth Dickason. “Looking Back: Grace Murray Hopper’s Younger Years.” *Chips* 12 no. 2 (April 1992).

[2] *Ibid*.

[3] http://www.aya.yale.edu/grad/wilburcross/bydept.htm Association of Yale Alumni: Graduate School Alumni. *Grace Murray Hopper was the first woman ever to achieve a PhD in mathematics from Yale.

[4] Dickason, Elizabeth. “Looking Back: Grace Murray Hopper’s Younger Years.”

[5] JG means “Junior Grade” a Lieutenant (JG) is the Navy equivalent of an Army or Marine Corps First Lieutenant in the United States Armed Forces.

[6] http://www-03.ibm.com/ibm/history/exhibits/markI/markI_intro.html *IBM’s ASCC introduction*

[7] Kathleen Broome Williams. *Grace Hopper: Admiral of the Cyber Sea*. (Naval Institute Press: Annapolis, 2004), 26.

[8] *Ibid*. 45-51

[9] *Ibid*. 50-51

[10] *Ibid*. 53

[11] Elizabeth Dickason

[12] *Ibid*. 70

[13] *Ibid*. 75-77

[14] *Ibid*. 77

[15] *Ibid.* 80

[16] Grace Hopper. "The Education of a Computer." *Proceedings*: *Symposium on Industrial Applications of Automatic Computing Equipment*. (Midwest Research Institute: Kansas City, 1953), 8-9

[17] Williams, 80.

[18] *Ibid*. 83

[19] *Ibid. 85*

[20] *Ibid*. 87

[21] *Ibid*. 87

[22] *Ibid*. 92

[23] COBOL has undergone many updates through the years and versions of it have been updated to keep pace with innovations in desktop computing. It is used in many businesses to this day to keep track of things like employee paychecks, or sales records.

[24] *Ibid*. 93-94

[25] *Ibid*. 95

[26] "Transcript of Naval Service for Commodore Grace Murray Hopper, US Naval Reserve," 1 July 1985, Grace Hopper file, Biographies, 20th Century, Navy Department Library

[27] Williams. 98, 104, 113

[28] The ranks of Commodore and Admiral were merged to create Rear Admiral (lower-half) and Rear Admiral (upper-half) followed by the highest rank in the modern U.S. Navy, Admiral. Grace Hopper held the rank of Rear Admiral (lower-half) at the time of her retirement in 1986.

[29] "Biographies in Naval History: Admiral Hyman G. Rickover." 25 July 2001 http://www.history.navy.mil/bios/rickover.htm (16 October 2001).

[30] "Transcript of Naval Service for Commodore Grace Murray Hopper, US Naval Reserve," 1 July 1985, Grace Hopper file, Biographies, 20th Century, Navy Department Library.

[31] Charles W. Billings. *Grace Hopper: Navy Admiral and Computer Pioneer.* (Enslow Publishers, Inc.: USA, 1989) 64.

[32] Charles W. Billings. *Grace Hopper: Navy Admiral and Computer Pioneer.* (Enslow Publishers, Inc.: USA, 1989) 57.